Reinforcement Learning: An Introduction

Solutions: Chapter 4

Mrinank Sharma

March 30, 2020

Exercise 4.1

$$q_\pi(11, \texttt{down}) = -1 + v_\pi(\texttt{terminal}) = -1. \tag{1}$$
$$q_\pi(7, \texttt{down}) = -1 + v_\pi(11) = -1 - 14 = -15. \tag{2}$$

Exercise 4.2: MDP Modifications

Unchanged dynamics from current states. In this case, the value of these states from these policies is unchanged, we've simply introduced a new state. We can calculate the value of this state using the Bellman equations:

$$v_\pi(15) = -1 + \frac{1}{4}[-22 - 20 - 14 + v_\pi(15)] \Rightarrow v_\pi(15) = -20. \tag{3}$$

Dynamics changed from state 13. In general, we'd expect this to change the value of state 13, which would propagate to all of the other states, meaning that we'd have to recalculate the values. Let's pretend that we were running iterative policy evaluation using the previous value function as our initialisation. Let's update the value for state 13:

$$v_\pi(13) = -1 + \frac{1}{4}[-20 - 22 - 14 \quad \underbrace{-20}_{\text{Estimate for } v_\pi(15)} \quad] = -20 \tag{4}$$

The value of state 13 **hasn't changed!** Therefore, the value of state 15, and all of the other states also won't change. We've recalculated values without having to solve those annoying equations, which is nice.

Exercise 4.3: Iterative Action-Value Evaluation

$$\begin{aligned} q_\pi(s,a) &= \mathbb{E}_\pi[G_t | S_t = s, A_t = a] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a] \\ &= \sum_{s' \in \mathcal{S}, r \in \mathcal{R}} p(s', r | s, a)[r + \gamma \sum_{a \in \mathcal{A}(s')} \pi(a'|s')[q_\pi(s', a')]], \end{aligned} \tag{5}$$

which can straightforwardly be turned into an iterative update equation.

Exercise 4.4: Policy Iteration Bug

If policy is switching between policies which are equally good, both of the policies are optimal. There must be multiple actions in each step which give the same expected optimal return, and the suggested algorithm does not state how to break ties. We should be able to fix this by settling ties in the arg max operation, for example, by indexing all of the actions and always choosing the lowest index amongst optimal options.

Algorithm 1 Policy Iteration

Initialisation: initialise $V(s)$ arbitarily for all $s \in \mathcal{S}$ except that $V(\text{terminal}) = 0$. Initalise $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$.
Parameters: $\theta > 0$, a threshold determining accuracy.

loop: ▷ Policy Evaluation
  $\Delta \leftarrow 0$
  for each $s \in \mathcal{S}$ **do**:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_{a \in \mathcal{A}(s)} \pi(a|s) \sum_{s' \in \mathcal{S}, r \in \mathcal{R}} p(s', r|s, a)[r + \gamma V(s')]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

`policy-stable` $\leftarrow$ True ▷ Policy Improvement
for each $s \in \mathcal{S}$ **do**:
  `old-action` $\leftarrow \pi(s)$
  $\pi(s) \leftarrow \arg\max_{a \in \mathcal{A}(s)} \sum_{s' \in \mathcal{S}, r \in \mathcal{R}} p(s', r|s, a)[r + \gamma V(s')]$ ▷ Settle ties by index.
  if `old-action` $\neq \pi(s)$ **then**:
    `policy-stable` $\leftarrow$ True
if `policy-stable` **then**:
  return $V \simeq v_*, \pi \simeq \pi*$
else:
  go to 1

Exercise 4.5: Policy Iteration with $q(s, a)$

I will assume deterministic policies here, using the same bug fix.

Algorithm 2 Q Iteration

Initialisation: initialise $Q(a, s)$ arbitarily for all $s \in \mathcal{S}, a \in \mathcal{A}$ except that $Q(\text{terminal}, a) = 0 \quad \forall a$. Initalise $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$.
Parameters: $\theta > 0$, a threshold determining accuracy.

loop: ▷ Policy Evaluation
  $\Delta \leftarrow 0$
  for each $s \in \mathcal{S}$ **do**:
    for each $a \in \mathcal{A}(s)$ **do**
      $q \leftarrow Q(s, a)$
      $Q(s, a) \leftarrow \sum_{s' \in \mathcal{S}, r \in \mathcal{R}} p(s', r|s, a)[r + \gamma Q(s', \pi(s'))]$
      $\Delta \leftarrow \max(\Delta, |q - Q(s, a)|)$
until $\Delta < \theta$

`policy-stable` $\leftarrow$ True ▷ Policy Improvement
for each $s \in \mathcal{S}$ **do**:
  `old-action` $\leftarrow \pi(s)$
  $\pi(s) \leftarrow \arg\max_{a \in \mathcal{A}(s)} Q(s, a)$ ▷ Settle ties by index.
  if `old-action` $\neq \pi(s)$ **then**:
    `policy-stable` $\leftarrow$ True
if `policy-stable` **then**:
  return $Q \simeq q_*, \pi \simeq \pi*$
else:
  go to 1

Exercise 4.6: ϵ-soft policies

Let's restrict ourself to determinstic + ϵ-soft policies. i.e.,

$$\pi(a|s) = \begin{cases} 1 - \frac{|\mathcal{A}(s)|-1}{|\mathcal{A}(s)|}\epsilon, & a = \pi_d(s) \\ \frac{\epsilon}{\mathcal{A}(s)}, & \text{otherwise} \end{cases}, \tag{6}$$

where π_d is now effectively the determinstic part of the policy. Now, we make the following changes:

1. No change to the initialise of the algorithm.
2. Update the value of the policy by adding a marginalisation step over the possible actions to be selected.
3. In policy improvement, update only the determinstic part of the policy, but using the value of the soft policy to perform the updates.

Note: not fully clear to me that considering this form is policy is sufficient. There are many stochastic policies which we could add a soft part to.